Classical Foundations of Algorithms for Solving Positive Definite Toeplitz Equations *

Gregory S. Ammar[†]

Abstract

The ongoing development and analysis of efficient algorithms for solving positive definite Toeplitz equations is motivated to a large extent by the importance of these equations in signal processing applications. The role of positive definite Toeplitz matrices in this and other areas of mathematics and engineering stems from Schur's study of bounded analytic functions on the unit disk, and Szegő's theory of polynomials orthogonal on the unit circle. These ideas underlie several Toeplitz solvers, and provide a useful framework for understanding the relationships among these algorithms. In this paper we give an overview of several direct algorithms for solving positive definite Toeplitz systems of linear equations from this classical viewpoint.

1 Introduction

Problems related to solving the system of equations $Mx = b$, where

$$M = M_n = \begin{bmatrix} \mu_0 & \mu_{-1} & \mu_{-2} & \cdots & \mu_{1-n} \\ \mu_1 & \mu_0 & \mu_{-1} & \ddots & \vdots \\ \mu_2 & \mu_1 & \mu_0 & \ddots & \mu_{-2} \\ \vdots & \ddots & \ddots & \ddots & \mu_{-1} \\ \mu_{n-1} & \cdots & \mu_2 & \mu_1 & \mu_0 \end{bmatrix} = [\mu_{i-j}]_{i,j=0}^{n-1}$$

is a Hermitian positive definite Toeplitz matrix, arise in several fundamental problems of signal processing, time series analysis, and image processing. There has therefore been a significant amount of effort devoted to developing efficient algorithms for solving positive definite Toeplitz equations. We refer to any of these algorithms as a *Toeplitz solver.*

*To appear in *Calcolo.* Presented at the Workshop on Toeplitz Matrices: Structure, Algorithms and Applications, Cortona, Italy, September 11, 1996.

†Department of Mathematical Sciences, Northern Illinois University, DeKalb, IL 60115 USA. Email: ammar@math.niu.edu.

The development of Toeplitz solvers goes back to the work of Levinson [33] in 1947, who presented a fast algorithm for solving a positive definite Toeplitz system of equations for solving discrete-time Wiener filtering problems. This work and subsequent work by Durbin [17] for solving the Yule-Walker equations of prediction theory, Trench [37] for constructing M^{-1} from the solution of one set of Yule-Walker equations, and Bareiss [8] for computing a triangular factorization of M form the foundation for direct algorithms for Toeplitz equations.

Toeplitz solvers can be divided into two classes: direct methods and iterative methods. The direct methods are further classified according to their arithmetic complexity. We will say that an algorithm that requires $O(n^2)$ floating-point arithmetic operations (flops) is a *fast* algorithm. This is in contrast with the $O(n^3)$ flops required by algorithms for arbitrary positive definite matrices, such as Cholesky's method. We will also refer to a method that requires $O(n \log^2 n)$ flops as a *superfast* algorithm. While any superfast algorithm will require fewer flops than a fast algorithm for sufficiently large n, the practical value of a superfast algorithm will depend on when this *crossover* of flop counts is achieved, and also on the tractability of implementing the algorithm on a computer.

In this paper we give an overview of several direct Toeplitz solvers and their connections with Schur functions and Szegő polynomials. Although knowledge of the connections with the classical theory is not necessary to derive the algorithms, we find the connections illuminating on the overall mathematical structure of the problems being considered. In fact, we have found the classical connections invaluable in understanding and implementing the superfast Toeplitz solver described in [5, 6].

2 Schur's Algorithm and the Carathéodory–Toeplitz Theorem

The original mathematical importance of positive definite Toeplitz matrices is in the solution of the classical *Carathéodory coefficient problem*, proved independently by Toeplitz and Carathéodory in 1911. An analytic mapping $w = f(z)$ of the unit disk $|z| < 1$ into the closed right half-plane $\Re(w) \geq 0$ is said to be a *Carathéodory function*, or a *function in the class C*.

Theorem 2.1 (Carathéodory-Toeplitz Theorem) *The power series*

$$w = f(z) = \mu + \sum_{j=1}^{\infty} \mu_j z^j$$

defines a function in the class C if and only if the Hermitian Toeplitz matrix $M_n = [\mu_{j-k}]_{j,k=0}^{n-1}$ *is nonnegative definite for each integer* $n \geq 1$*, where* $\mu_0 = \mu + \overline{\mu}$ *and* $\mu_{-j} \equiv \overline{\mu}_j$ *for* $j > 0$*.*

In 1917, Schur gave a *constructive* proof of the Carathéodory-Toeplitz theorem via his study of analytic functions bounded in the unit disk [35]. In particular, Schur gave a procedure for determining when a given function $w = \phi(z)$ maps $|z| < 1$ analytically into $|w| \leq 1$. Such a function is now called a *Schur function*, or a function in the class S.

Schur showed that the function class S can be parameterized by certain sequences of complex numbers γ_j, known as *Schur parameters*. They determine a continued fraction representation of the given Schur function $\phi_0(z)$. Schur gave the following procedure for constructing these parameters.

Algorithm 2.1 (Schur's Algorithm)

Input: A complex-valued function $\phi(\lambda)$.

$\gamma_1 := \phi_0(0)$
for $k = 1, 2, \ldots$ **while** $|\gamma_k| < 1$

$$\phi_k(\lambda) := \frac{1}{\lambda}\,\frac{\phi_{k-1}(\lambda) - \gamma_k}{1 - \overline{\gamma_k}\phi_{k-1}(\lambda)}$$

$\gamma_{k+1} := \phi_k(0)$
end.

The initial function ϕ_0 is a Schur function if and only if one of the following holds:

(1) $|\gamma_k| < 1$ for $k = 1, 2, \ldots$;

(2) $|\gamma_k| < 1$ for $k = 1, 2, \ldots, n-1$, $|\gamma_n| = 1$, and $\phi_n(\lambda) \equiv \gamma_n$.

If condition (2) holds, then ϕ_0 is a rational Schur function.

It is natural to view the Schur functions generated by the algorithm as ratios of formal power series,

$$\phi_j(\lambda) = \frac{\alpha_j(\lambda)}{\beta_j(\lambda)} = \frac{\alpha_{j0} + \alpha_{j1}\lambda + \alpha_{j2}\lambda^2 + \cdots}{\beta_{j0} + \beta_{j1}\lambda + \beta_{j2}\lambda^2 + \cdots}.$$

In terms of these power series, we can restate Schur's algorithm as

$$\alpha_n(\lambda) = \frac{1}{\lambda}\big(\alpha_{n-1}(\lambda) - \gamma_n\beta_{n-1}(\lambda)\big) \qquad \beta_n(\lambda) = \beta_{n-1}(\lambda) - \overline{\gamma}_n\alpha_{n-1}(\lambda) \tag{2.1}$$

If we partially normalize the initial ratio so that $\beta_{0,0} > 0$, then the constant terms of the denominators are related by

$$\beta_{j,0} = (1 - |\gamma_j|^2)\beta_{j-1,0}$$

so that $\phi_0(\lambda)$ is a Schur function if and only if $\{\beta_{j,0}\}$ is a nonnegative, nonincreasing sequence that is infinite, or finite and terminating with $\beta_{n,0} = 0$.

Schur gave explicit determinantal formulas for the numerators and denominators in (2.1), and using the the correspondence

$$\alpha_0(z) = \sum_{j=0}^{\infty} \alpha_{0j} z^j \leftrightarrow A_\infty := \begin{bmatrix} \alpha_{00} & 0 & 0 & \cdots \\ \alpha_{01} & \alpha_{00} & 0 & \cdots \\ \alpha_{02} & \alpha_{01} & \alpha_{00} & \ddots \\ \vdots & \ddots & \ddots & \ddots \end{bmatrix}.$$

between the algebra of formal power series and the algebra of singly infinite triangular Toeplitz matrices, he concluded that

$$\beta_{n0} = \det(B_n B_n^H - A_n A_n^H).$$

where A_n and B_n are the $n \times n$ leading principal submatrices of A_∞ and B_∞. This gives the following characterization of Schur functions.

Theorem 2.2 ([35]) *The ratio of formal power series $\phi_0(z) = \alpha_0(z)/\beta_0(z)$ is in the class S if and only if the matrix*

$$B_n B_n^H - A_n A_n^H$$

is nonnegative definite for each integer $n \geq 1$.

The Carathéodory-Toeplitz theorem follows directly from this result. In particular, the power series $f(z)$ of Theorem 1 is in the class C if and only if

$$\frac{1 - f(z)}{1 + f(z)} =: \frac{\alpha_0(z)}{\beta_0(z)} = \phi_0(z)$$

is in the class S. By Theorem 2.2, this is true if and only if the matrix

$$(I + F_n)(I + F_n)^H - (I - F_n)(I - F_n)^H = 2(F_n + F_n^H) =: 2M_n$$

is nonnegative definite for every n.

3 Fast Cholesky factorization

In 1969, Bareiss [8] presented an algorithm of complexity $O(n^2)$ for computing a triangular factorization of a Toeplitz matrix. When applied to a positive definite Toeplitz matrix M $= M_{n+1}$, Bareiss's algorithm computes the Cholesky factorization

$$M =: LDL^H, \tag{3.1}$$

where L is a unit lower triangular matrix, and $D = diag[\delta_0, \delta_1, \ldots \delta_n]$, with each $\delta_j > 0$.

It is now well known that the algorithm of Bareiss applied to a positive definite Toeplitz matrix is a manifestation of Schur's algorithm. Algorithms related to Bareiss's algorithm are therefore also referred to as algorithms of 'Schur type'. See [28, 31] for discussions of this connection as well as wider connections of Schur's algorithm with signal processing and matrix computation. The explicit relationship between Schur's algorithm and Cholesky factorization is given in the following proposition [5].

Proposition 3.1 *Let M_{n+1} be a Hermitian positive definite Toeplitz matrix. Then the rational function*

$$\phi_0(\lambda) = \frac{\alpha_0(\lambda)}{\beta_0(\lambda)} := \frac{\sum_{j=0}^{n-1} -\bar{\mu}_{j+1}\lambda^j}{\sum_{j=0}^{n} \bar{\mu}_j \lambda^j} \tag{3.2}$$

is a Schur function. Moreover, the entries of the lower triangular matrix $LD = [\tau_{j,k}]_{j,k=0}^n$ are given by $\tau_{j,k} = \overline{\beta}_{k,j-k}$ for $k \leq j$, where $\beta_k(\lambda)$ is the denominator polynomial that results from k steps of Schur's algorithm applied to $\alpha_0(\lambda)$, $\beta_0(\lambda)$ given in (3.2).

Thus, the jth column of the Cholesky factor L is determined directly from the first $n-j$ coefficients of the denominator polynomial β_j of each function ϕ_j generated by Schur's algorithm.

The computations involved in Schur's algorithm can be viewed in terms of infinitely long column vectors containing the coefficients of $\beta_j(\lambda)$ and $\alpha_j(\lambda)$. The Schur parameter $\gamma_{j+1} = \alpha_{j,0}/\beta_{j,0}$ determines a scaled hyperbolic transformation that produces a zero in the first entry of the second column. Then the second column is shifted up, or equivalently, the first column is shifted down, corresponding to dividing the numerator by λ.

$$\begin{bmatrix} \beta_{0,0} & \alpha_{0,0} \\ \beta_{0,1} & \alpha_{0,1} \\ \beta_{0,2} & \alpha_{0,2} \\ \vdots & \vdots \\ \beta_{0,n} & \alpha_{0,n} \\ \vdots & \vdots \end{bmatrix} \begin{bmatrix} 1 & -\gamma_1 \\ \overline{\gamma}_1 & 1 \end{bmatrix} = \begin{bmatrix} \beta_{1,0} & 0 \\ \beta_{1,1} & \alpha_{1,0} \\ \beta_{1,2} & \alpha_{1,1} \\ \vdots & \vdots \\ \beta_{1,n} & \alpha_{1,n-1} \\ \vdots & \vdots \end{bmatrix} \longrightarrow \begin{bmatrix} \beta_{1,0} & \alpha_{1,0} \\ \beta_{1,1} & \alpha_{1,1} \\ \vdots & \vdots \\ \beta_{1,n-1} & \alpha_{1,n-1} \\ \vdots & \vdots \end{bmatrix}$$

Of course, in practice only finitely many of the coefficients α_{0k} and β_{0k} will be used. Note that the first n coefficients of α_0 and β_0 will determine the first $n-k$ coefficients of α_k and β_k ($k = 1, 2, \ldots, n$) and the first n Schur parameters γ_k. One can arrange the computation of these coefficients to proceed 'column-by-column', each step producing vectors of length one less than the previous column. We refer to this arrangement as the *parallel form* of Schur's algorithm, since parallel implementations of Schur's algorithm will produce one set of columns from the previous set in one time step. This it is the natural form for the algorithm from the perspective of *displacement structure* (see [31]).

Alternately, the computations can be arranged so that each pair of coefficients $(\alpha_{0,j}, \beta_{0,j})$ of the initial power series is entered and processed sequentially. This results in the *progressive form* of Schur's algorithm which, in terms of the infinite column vectors above, can be thought of as a rowwise arrangement of Schur's algorithm. This form is more natural from the point of view of continued fractions, since the number of coefficients in the initial numerator and denominator need not be known in advance. In fact, having generated the nth row of LD, and the first n Schur parameters, one can naturally add the next row, and generate the next Schur parameter.

Algorithm 3.1 (Progressive Schur Algorithm)

$$
\begin{array}{l}
for\ k = 1, 2, 3, \ldots n \\
\quad read\ \alpha_{0,k-1}\ and\ \beta_{0,k-1} \\
\quad for\ j = 1, 2, 3, \ldots, k-1 \\
\qquad \begin{bmatrix} \alpha_{j,k-j-1} \\ \beta_{j,k-j} \end{bmatrix} = \begin{bmatrix} 1 & -\gamma_j \\ -\bar{\gamma}_j & 1 \end{bmatrix} \begin{bmatrix} \alpha_{j-1,k-j} \\ \beta_{j-1,k-j} \end{bmatrix} \\
\qquad \gamma_k = \alpha_{k-1,0}/\beta_{k-1,0}, \\
\qquad \beta_{k,0} = \beta_{k-1,0}(1 - |\gamma_k|^2)
\end{array}
$$

The parallel and progressive forms of Schur's algorithm are simply rearrangements of each other. They compute the Schur parameters γ_k and the first $(n-k)$ coefficients of α_k and β_k, $k = 1, 2, \ldots, n$, using roughly n^2 multiplications and n^2 additions.

4 The Szegő Recursions and Levinson's Algorithm

Another class of Toeplitz solvers arises from Szegő's theory of polynomials that are orthogonal with respect to a measure on the unit circle in the complex plane. A bounded nondecreasing distribution function $\alpha(t)$ with more than n points of increase on $[-\pi, \pi]$ defines a measure on the unit circle on the complex plane, and an inner product $(\cdot, \cdot)$ on the set $\mathbb{C}_n[\lambda]$ of polynomials of degree at most n, according to

$$(\alpha(\lambda), \beta(\lambda)) = \int_{-\pi}^{\pi} \overline{\alpha(e^{it})}\beta(e^{it})d\alpha(t). \tag{4.1}$$

This inner product on $\mathbb{C}_n[\lambda]$ is determined by its moments (λ^j, λ^k), $0 \le j, k \le n$. Note that

$$(\lambda^j, \lambda^k) = (\lambda^j \overline{\lambda}^k, 1) = (\lambda^{j-k}, 1) =: \mu_{j-k},$$

so that the (j,k) moment depends only on the difference $j - k$. The *moment matrix* $M_{n+1} = [\mu_{j-k}]_{j,k=0}^{n}$ is therefore a Hermitian positive definite Toeplitz matrix. Conversely, it follows from the solution of the classical trigonometric moment problem that every positive definite Hermitian Toeplitz matrix M_{n+1} arises as the first $n+1$ moments of an inner product on the unit circle in this manner. See, for example, [1, 25].

The unique family of monic polynomials $\{\psi_j(\lambda)\}_{j=0}^{n}$ that satisfy

$$(\psi_j, \psi_k) = \begin{cases} 0 & \text{if } j \neq k, \\ \delta_j > 0 & \text{if } j = k. \end{cases} \tag{4.2}$$

with $\deg(\psi_j(\lambda)) = j$ is said to be *orthogonal with respect to a measure on the unit circle.* These orthogonal polynomials were studied by Szegő [36] and are also known as the monic *Szegő polynomials* associated with M_{n+1}. Szegő showed that these polynomials satisfy a simple recurrence relation, from which it follows that they can be generated using the following recursive procedure, which we refer to as the *Szegő recursions.*

$$\begin{array}{l} \psi_0(\lambda) := 1, \quad \delta_0 := \mu_0, \\ \textbf{for } j = 0, 1, \ldots, n-1 \textbf{ do} \\ \left\lfloor \begin{array}{l} \gamma_{j+1} := -(1, \lambda\psi_j)/\delta_j \\ \psi_{j+1}(\lambda) := \lambda\psi_j(\lambda) + \gamma_{j+1}\tilde{\psi}_j(\lambda) \\ \delta_{j+1} := \delta_j\big(1 - |\gamma_{j+1}|^2\big), \end{array} \right. \end{array} \tag{4.3}$$

where $\tilde{\psi}_j(\lambda) = \lambda^j \overline{\psi}_j(1/\lambda)$ denotes the polynomial obtained by conjugating and reversing the coefficients of $\tilde{\psi}_j$ in the monomial basis of $\mathbb{C}_j[\lambda]$.

The coefficients of the monic Szegő polynomials provide a factorization of the inverse of the Toeplitz matrix. In particular, let $R_{n+1} = [\rho_{j,k}]_{j,k=0}^{n}$ denote the unit right triangular matrix of order $n+1$ whose kth column contains the coefficients of $\psi_k(\lambda) = \sum_{j=0}^{k} \rho_{j,k}\lambda^j$, with $\rho_{k,k} = 1$ and $\rho_{j,k} = 0$ for $j > k$, $0 \leq k \leq n$. Then in terms of matrices, the orthogonality condition (4.2) becomes

$$R_{n+1}^H M_{n+1} R_{n+1} = D_{n+1}, \tag{4.4}$$

or equivalently,

$$M_{n+1}^{-1} = R_{n+1} D_{n+1}^{-1} R_{n+1}^H, \tag{4.5}$$

where $D_{n+1} := diag[\delta_k]_{k=0}^{n}$. Thus, the coefficients of the Szegő polynomials determine the *reverse Cholesky factorization* of M^{-1}, and moreover, the Szegő recursions provide an $O(n^2)$ algorithm for constructing this factorization. This algorithm is known as the *Levinson-Durbin algorithm.* See, for example, [10, 24].

The Cholesky factorization (3.1) and inverse Cholesky factorization (4.4) of M are related by $L^{-1} = R^H$, so that there is a close connection between Schur's algorithm and the Levinson-Durbin algorithm. In fact, when ϕ_0 is defined from M_{n+1} as in Proposition 3.1, then the Schur parameters generated by Schur's algorithm are equal to the recurrence coefficients $\{\gamma_j\}_{j=1}^{n}$ generated during the Szegő recursions. These parameters are often of interest in applications, where they are known as *reflection coefficients* or *partial correlation coefficients.*

This gives rise to the possibility of a hybrid algorithm, in which the Szegő polynomials are generated by the Szegő recursions, using recurrence coefficients $\{\gamma_j\}_{j=1}^n$ generated by Schur's algorithm. Such an algorithm for computing Szegő polynomials is proposed in [20] because of its advantages in parallel computing. Moreover, Schur's algorithm has better numerical behavior than Levinson's algorithm [21, 11], so one can expect the hybrid approach to produce more accurate Szegő polynomial coefficients than the Levinson-Durbin algorithm [21].

5 Toeplitz Inversion Formulas

Direct Toeplitz solvers can be divided into two phases: a factorization or decomposition phase and a solution phase. Toeplitz solvers that rely on triangular factorizations of M or of M^{-1} require $O(n^2)$ flops in the first phase to compute the factorization, and an additional $O(n^2)$ flops in the second phase to use the factorization to solve $Mx = b$. However, several formulas exist for expressing M^{-1} in terms of its last column (or equivalently, in terms of ψ_n and δ_n) which can provide for increased efficiency in the second phase of a Toeplitz solver. These *Toeplitz inversion formulas* also follow directly from results of Szegő.

The *reproducing kernel polynomial* $\kappa_n(\lambda, \tau)$ of degree n associated with the inner product defined by M_{n+1} is given by

$$\kappa_n(\lambda, \tau) = \sum_{j=0}^{n} \frac{\psi_j(\lambda)\overline{\psi_j(\tau)}}{\delta_j}. \tag{5.1}$$

Szegő proved the following identity:

$$\kappa_n(\lambda, \tau) = \frac{\tilde{\psi}_n(\lambda)\overline{\tilde{\psi}_n(\tau)} - \lambda\overline{\tau}\psi_n(\lambda)\overline{\psi_n(\tau)}}{\delta_n(1 - \lambda\overline{\tau})} \tag{5.2}$$

See [36, §11.4] or [25, §2.3]. The kernel polynomial is therefore determined by the Szegő polynomial of degree n and its squared norm δ_n. This formula is analogous to the Christoffel-Darboux formula for polynomials that satisfy a Jacobi-type three-term recurrence relation [36, 1].

By comparing (5.1) and (4.5), we see that the coefficient of $\lambda^j\overline{\tau}^k$ in $\kappa_n(\lambda, \tau)$ is the (j, k) entry of M_{n+1}^{-1}. Consequently, Szegő's formula (5.2) allows one to construct M_{n+1}^{-1} from the coefficients of the last Szegő polynomial $\psi_n(\lambda)$ and its squared norm δ_n. This was first observed by Trench [37], who presented an algorithm for constructing M^{-1} using $O(n^2)$ operations. See [10, 24] for concise descriptions of Trench's algorithm.

If we view the polynomials in Szegő's formula (5.2) in terms of the corresponding triangular Toeplitz matrices, we obtain a decomposition of M^{-1} called the *Gohberg-Semencul formula*. Specifically, let $r = [\rho_j]_{j=0}^n$, with $\rho_n = 1$, denote the vector containing the coefficients of the monic Szegő polynomial $\psi_n(\lambda)$, and let $L(x)$ denote the lower triangular Toeplitz matrix whose first column is $x \in \mathbb{C}^{n+1}$. Also let $Z_{n+1} = L(e_1)$ and $J_{n+1} = [e_n, e_{n-1}, \ldots, e_0]$ denote the *downshift matrix* and the *reversal matrix*, respectively, where $[e_0, e_1, \ldots, e_n] = I_{n+1}$ is the identity matrix of order $n+1$. Then the vectors $r_0 = Z_{n+1}r$ and $r_1 = J_{n+1}\overline{r}$ correspond to $\lambda\psi_n(\lambda)$ and $\tilde{\psi}_n(\lambda)$, respectively, and we obtain from (5.2) the *Gohberg-Semencul formula* for M_{n+1}^{-1},

$$\delta_n M_{n+1}^{-1} = L(r_1)L(r_1)^H - L(r_0)L(r_0)^H. \tag{5.3}$$

This formula is actually one of several formulas introduced by Gohberg and Semencul that express the inverse of a Toeplitz matrix as the difference of products of triangular Toeplitz matrices [23, 26]. These formulas have had a significant impact in computational and theoretical problems involving Toeplitz matrices, and in generalizing algorithms for Toeplitz matrices to larger classes of matrices. One notable area of research stimulated by the Gohberg-Semencul formulas is the work on *displacement structures* of matrices, beginning with [30, 18]. These ideas have proved to be powerful in generalizing the Gohberg-Semencul formula and algorithms for Toeplitz equations to larger classes of matrices. See, for example, [32, 19, 29] as well as the recent survey [31] and references therein.

The utility of the Gohberg-Semencul formula in the solution of Toeplitz equations stems from the fact that it can be used to compute $M_{n+1}^{-1}b$ using fast Fourier transform (FFT) techniques in $O(n \log n)$ arithmetic operations [27]. This is especially advantageous when several systems of equations involving the same Toeplitz matrix are to be solved, since the first phase of the Toeplitz solver, which involves a higher order amount of computation, would be performed only once.

Toeplitz inversion formulas that provide for increased computational efficiency in the second phase of a Toeplitz solver are presented in [3, 2], in which triangular Toeplitz matrices in (5.3) are replaced with circulant and skew-circulant matrices determined by the vector r. These formulas were derived using ideas of *cyclic displacement* from [19], but they can also be derived directly from the Gohberg-Semencul formula. The formula involving circulant and skew-circulant matrices presented in [2] provides computational savings of over 42 percent relative to the use of the Gohberg-Semencul formula in the second phase of a Toeplitz solver. Since then there have been several other formulas presented that achieve this same level of efficiency in multiplying a vector by the inverse of a Toeplitz matrix, and also generalize to a variety of other matrix structures [22, 12, 16, 13].

6 The Generalized Schur Algorithm

Direct algorithms of arithmetic complexity $O(n \log^2 n)$ for performing Phase 1 of a Toeplitz solver have been presented in [14, 9, 15, 34, 5, 6]. These algorithms all rely on Toeplitz inversion formulas for their second phase, and use *doubling procedures* and fast Fourier transform techniques to perform the first phase. We refer to these as algorithms *superfast Toeplitz solvers.* However, these algorithms will involve fewer arithmetic operations than a fast algorithm only for sufficiently large n. It is shown in [5] that the superfast Toeplitz solver presented independently by Musicus [34] and de Hoog [15] can be interpreted in terms of a doubling strategy applied to the continued fraction generated by Schur's algorithm. This algorithm can be viewed as a hybrid algorithm that uses the results of a fast implementation of Schur's algorithm to construct the Szegő polynomial ψ_n.

6.1 Schur polynomials.

The functions ϕ_k generated by Schur's algorithm are related by

$$\phi_{k-1}(\lambda) = t_k\big(\phi_k(\lambda)\big)$$

where t_k denotes the elementary linear fractional transformation (LFT)

$$t_k(\tau) = \frac{\gamma_k + \lambda\tau}{1 + \bar{\gamma}_k \lambda\tau}.$$

We therefore have $\phi_0 = T_k(\phi_k)$, where T_k is the composition

$$T_k(\tau) = t_1 \circ t_2 \circ \cdots \circ t_k(\tau).$$

In this way we obtain the *Schur continued fraction representation* of ϕ_0,

$$\phi_0 = T_n(\phi_n) = t_1 \circ t_2 \circ \ldots \circ t_n(\phi_n),$$

The function $T_n(0)$ is referred to as the *nth approximant* of ϕ_0, and ϕ_n is called the *nth tail* of ϕ_0.

It is shown in [5], and easily verified, that the LFT $T_n(\tau)$ can be expressed as

$$T_n(\tau) = \frac{\xi_n + \tilde{\eta}_n\tau}{\eta_n + \tilde{\xi}_n\tau},$$

where the polynomials ξ_n and η_n satisfy the recurrence relations

$$\begin{bmatrix} \tilde{\eta}_n & \xi_n \\ \tilde{\xi}_n & \eta_n \end{bmatrix} = \begin{bmatrix} \tilde{\eta}_{n-1} & \xi_{n-1} \\ \tilde{\xi}_{n-1} & \eta_{n-1} \end{bmatrix} \begin{bmatrix} \lambda & \gamma_n \\ \overline{\gamma}_n\lambda & 1 \end{bmatrix}, \qquad \begin{bmatrix} \tilde{\eta}_0 & \xi_0 \\ \tilde{\xi}_0 & \eta_0 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}. \tag{6.1}$$

It follows by induction that ξ_n and η_n have degree *less than* n, and $\tilde{\xi}_n(\lambda) := \lambda^n\bar{\xi}_n(1/\lambda)$, $\tilde{\eta}_n(\lambda) := \lambda^n\bar{\eta}_n(1/\lambda)$, $\xi_n(0) = \gamma_n$, and $\eta_n(0) = 1$ for all n.

The polynomials ξ_n and η_n are generalizations of the Schur parameters in the sense that they determine a composition of the elementary linear fractional transformations determined by the Schur parameters. We will refer to ξ_n and η_n as the nth *Schur polynomials* associated with the Schur function ϕ_0.

The following result shows how the Schur polynomials provide the Szegő polynomial ψ_n, and hence provide for the first phase of a Toeplitz solver.

Proposition 6.1 *Let M, α_0, β_0 be as in Proposition 3.1, and let ξ_k, η_k be the kth Schur polynomials determined by α_0, β_0. Then the monic Szegő polynomial ψ_k corresponding with M is given by*

$$\bar{\psi}_k(\lambda) = \tilde{\eta}_k(\lambda) + \tilde{\xi}_k(\lambda)/\lambda. \tag{6.2}$$

The proof follows by comparison of the recurrence relations for ψ_k with those for ξ_k and η_k [5].

6.2 The Generalized Schur Algorithm

We can now describe the doubling generalization of Schur's algorithm, which forms the basis for the first phase of our superfast Toeplitz solver.

Recall that if we perform m steps of Schur's algorithm on the Schur function ϕ_0, we can obtain the mth Schur polynomials ξ_m and η_m. Moreover, these polynomials are determined by the first m coefficients of α_0 and β_0. Let $\alpha_0^{(m)}(\lambda)$ and $\beta_0^{(m)}(\lambda)$ denote the polynomials of degree less than m formed from these coefficients. To describe the doubling procedure, we assume that ξ_m and η_m have been computed from $\alpha_0^{(m)}$ and $\beta_0^{(m)}$, and we seek to compute ξ_{2m} and η_{2m} from $\alpha_0^{(2m)}$ and $\beta_0^{(2m)}$.

Having obtained ξ_m and η_m, the mth Schur function ϕ_m is given by

$$\phi_m = \frac{\alpha_m}{\beta_m} = T_m^{-1}(\phi_0) = \frac{\alpha_0\eta_m - \beta_0\xi_m}{\beta_0\tilde{\eta}_m - \alpha_0\tilde{\xi}_m}.$$

It can be shown [5] that

$$\begin{array}{rcl} \alpha_0\eta_m - \beta_0\xi_m & = & \beta_{0,0}\gamma_{m+1}\delta_m\lambda^m + O(\lambda^{m+1}) \\ \beta_0\tilde{\eta}_m - \alpha_0\tilde{\xi}_m & = & \beta_{0,0}\delta_m\lambda^m + O(\lambda^{m+1}), \end{array} \tag{6.3}$$

so that the first m terms of the numerator and denominator of ϕ_m can be taken from

$$\begin{array}{rcl} \alpha_m^{(m)}(\lambda) & = & (\alpha_0^{(2m)}(\lambda)\eta_m(\lambda) - \beta_0^{(2m)}(\lambda)\xi_m(\lambda))/\lambda^m, \\ \beta_m^{(m)}(\lambda) & = & (\beta_0^{(2m)}(\lambda)\tilde{\eta}_m(\lambda) - \alpha_0^{(2m)}(\lambda)\tilde{\xi}_m(\lambda))/\lambda^m. \end{array} \tag{6.4}$$

We can now perform the doubling step: Since ϕ_m is also a Schur function, we can use the same procedure that computed ξ_m and η_m from $\alpha_0^{(m)}$ and $\beta_0^{(m)}$ to compute the Schur polynomials $\xi_{m,m}$ and $\eta_{m,m}$ that result from m steps of Schur's algorithm applied to $\alpha_m^{(m)}$ and $\beta_m^{(m)}$.

Let $T_{m,m}$ denote the LFT that results from m steps of Schur's algorithm applied to ϕ_m. Then we have $\phi_0 = T_m(\phi_m) = T_m(T_{m,m}(\phi_{2m}))$, so that $T_{2m} = T_m \circ T_{m,m}$. Writing this composition in terms of Schur polynomials, we obtain

$$\begin{array}{rcl} \xi_{2m} & = & \tilde{\eta}_m\xi_{m,m} + \xi_m\eta_{m,m} \\ \eta_{2m} & = & \tilde{\xi}_m\xi_{m,m} + \eta_m\eta_{m,m}. \end{array} \tag{6.5}$$

This discussion is summarized by the following recursive description of the doubling procedure.

Algorithm 6.1 *To compute ξ_{2m}, η_{2n} from $\alpha_0^{(2m)}$, $\beta_0^{(2m)}$:*

Step 0: Compute ξ_m and η_m from $\alpha_0^{(m)}$ and $\beta_0^{(m)}$.

Step 1: Use (6.4) to compute $\alpha_m^{(m)}$ and $\beta_m^{(m)}$.

Step 2: (The doubling step.) Compute $\xi_{m,m}$ and $\eta_{m,m}$ from $\alpha_m^{(m)}$ and $\beta_m^{(m)}$ as ξ_m and η_m were obtained from $\alpha_0^{(m)}$ and $\beta_0^{(m)}$.

Step 3: Use (6.5) to compute ξ_{2m} and η_{2m} from ξ_m, η_m, $\xi_{m,m}$, and $\eta_{m,m}$.

The algorithm can be started by performing Step 0 directly by, for example, setting $\xi_1 = \alpha_{0,0}/\beta_{0,0}$ and $\eta_1 = 1$. More generally, we can use Schur's algorithm to generate $\{\gamma_j\}_{j=1}^{n_0}$ for a small value of n_0, and obtain ξ_{n_0}, η_{n_0} from the Schur parameters and the recursions (6.1).

Thus, the generalized Schur algorithm consists of various polynomial multiplications and additions performed in a recursive manner. The multiplication of polynomials can be efficiently performed using standard FFT techniques. This results in an $O(n \log^2 n)$ algorithm for computing ξ_n and η_n, where $n = 2^\nu$. The Toeplitz system of equations $M_{n+1}x = b$ can then be solved by forming ψ_n from ξ_n, η_n by Proposition 6.1, and using a Toeplitz inversion formula to perform the solution phase of the algorithm in $O(n \log n)$ arithmetic operations.

A detailed analysis of the implementation for Hermitian Toeplitz matrices is given in [4]. Further refinements for real Toeplitz matrices are described in [6], where it is shown in that the first phase of the superfast Toeplitz solver can be implemented using fewer than $8n \log_2^2 n$ real arithmetic operations for a real Toeplitz matrix M_{n+1}, where $n = 2^\nu$. Since the Levinson-Durbin algorithm requires more than $2n^2$ operations, the superfast algorithm has a smaller operation count for $n = 2^\nu \geq 256$. Experimental results presented in [7] confirm that the two procedures require approximately the same execution time for $n = 256$, and that the relative efficiency of the superfast algorithm quickly increases for larger values of $n = 2^\nu$. Moreover, experimental results indicate that there is little or no degradation in the accuracy of the superfast algorithm compared to the Levinson-Durbin algorithm [4, 7].

While the experimental results indicate that this superfast algorithm may be as reliable as the Levinson-Durbin algorithm for the first phase of a Toeplitz solver, no stability analysis along the lines of [21] or [11] has yet been performed. However, since the Schur polynomials are generalizations of the Schur parameters, the superfast algorithm is closely related to the hybrid algorithm mentioned at the end of Section 4. This is encouraging in that the superfast algorithm may behave more like the hybrid algorithm than the Levinson-Durbin algorithm.

References

[1] N. I. Akhiezer. *The Classical Moment Problem and Some Related Questions in Analysis*. Hafner, New York, 1965.

[2] G. S. Ammar and P. Gader. New decompositions of the inverse of a Toeplitz matrix. In M. A. Kaashoek, J. H. van Schuppen, and A. C. N. Ran, editors, *Signal Processing, Scattering and Operator Theory, and Numerical Methods*, pages 421–428. Birkhäuser, 1990.

[3] G. S. Ammar and P. Gader. A variant of the Gohberg–Semencul formula involving circulant matrices. *SIAM J. Matrix Anal. Appl.*, 12:534–540, 1991.

[4] G. S. Ammar and W. B. Gragg. Implementation and use of the generalized Schur algorithm. In C. I. Byrnes and A. Lindquist, editors, *Computational and Combinatorial Methods in Systems Theory*, pages 265–280. North-Holland, Amsterdam, 1986.

[5] G. S. Ammar and W. B. Gragg. The generalized Schur algorithm for the superfast solution of Toeplitz systems. In J. Gilewicz, M. Pindor, and W. Siemaszko, editors, *Rational Approximation and its Applications in Mathematics and Physics*, number 1237 in Lecture Notes in Mathematics, pages 315–330. Springer-Verlag, Berlin, 1987.

[6] G. S. Ammar and W. B. Gragg. Superfast solution of real positive definite Toeplitz systems. *SIAM J. Matrix Anal. Appl.*, 9:61–76, 1988.

[7] G. S. Ammar and W. B. Gragg. Numerical experience with a superfast real Toeplitz solver. *Linear Algebra and its Applications*, 121:185–206, 1989.

[8] E. H. Bareiss. Numerical solution of linear equations with Toeplitz and vector Toeplitz matrices. *Numer. Math.*, 13:404–424, 1969.

[9] R. R. Bitmead and B. D. O. Anderson. Asymptotically fast solution of Toeplitz and related systems of linear equations. *Lin. Alg. Appl.*, 34:103–116, 1980.

[10] R. E. Blahut. *Fast Algorithms for Digital Signal Processing*. Addison-Wesley, Reading, MA, 1985.

[11] A. W. Bojanczyk, R. P. Brent, F. R. D. Hoog, and D. R. Sweet. On the stability of the Bareiss and related Toeplitz factorization algorithms. *SIAM J. Matrix Anal. Appl.*, 16:40–57, 1995.

[12] E. Bozzo. Algebras of higher dimension for displacement decompositions and computations with Toeplitz plus Hankel matrices. *Lin. Alg. Appl.*, 230:127–150, 1995.

[13] E. Bozzo and C. Di Fiore. On the use of certain matrix algebras associated with real discrete transforms in matrix displacement decomposition. *SIAM J. Matrix Anal. Appl.*, 16:312–326, 1995.

[14] R. P. Brent, F. G. Gustavson, and D. Y. Y. Yun. Fast solution of Toeplitz systems of equations and computation of Padé approximants. *J. Algorithms*, 1:259–295, 1980.

[15] F. de Hoog. A new algorithm for solving Toeplitz systems of equations. *Lin. Alg. Appl.*, 88/89:123–138, 1987.

[16] C. Di Fiore and P. Zellini. Matrix decompositions using displacement rank and classes of commutative matrix algebras. *Lin. Alg. Appl.*, 229:49–100, 1995.

[17] J. Durbin. The fitting of time-series models. *Rev. Int. Inst. Statist.*, 28:233–243, 1960.

[18] B. Friedlander, M. Morf, T. Kailath, and L. Ljung. New inversion formulas for matrices classified in terms of their distance from Toeplitz matrices. *Lin. Alg. Appl.*, 27:31–60, 1979.

[19] P. Gader. Displacement operator based decompositions of matrices using circulants or other group matrices. *Lin. Alg. Appl.*, 139:111–131, 1990.

[20] I. Gohberg, T. Kailath, I. Koltracht, and P. Lancaster. Linear complexity parallel algorithms for linear systems of equations with recursive structure. *Lin. Alg. Appl.*, 88/89:271–315, 1987.

[21] I. Gohberg, I. Koltracht, and D. Xiao. Condition and accuracy of algorithms for computing Schur coefficients of Toeplitz matrices. *SIAM J. Matrix Anal. Appl.*, 15:1290–1309, 1994.

[22] I. Gohberg and V. Olshevsky. Complexity of multiplication with vectors for structured matrices. *Lin. Alg. Appl.*, 202:163–192, 1994.

[23] I. C. Gohberg and I. A. Fel′dman. *Convolution Equations and Projection Methods for Their Solution.* American Math. Soc., Providence, RI, 1974.

[24] G. H. Golub and C. F. Van Loan. *Matrix Computations.* Johns Hopkins University Press, second edition, 1989.

[25] U. Grenander and G. Szegő. *Toeplitz Forms and Their Applications.* Chelsea, New York, second edition, 1984.

[26] G. Heinig and K. Rost. *Algebraic Methods for Toeplitz-like Matrices and Operators.* Akademie-Verlag, Berlin, 1984.

[27] J. R. Jain. An efficient algorithm for a large Toeplitz set of linear equations. *IEEE Trans. Acoust., Speech, Signal Proc.*, 27:612–615, 1979.

[28] T. Kailath. A theorem of I. Schur and its impact on modern signal processing. In I. Gohberg, editor, *I. Schur Methods in Operator Theory and Signal Processing*, pages 9–30. Birkhäuser Verlag, Boston, 1986.

[29] T. Kailath and J. Chun. Generalized displacement structure for block-Toeplitz, Toeplitz-block, and Toeplitz-derived matrices. *SIAM J. Matrix Anal. Appl.*, 15(1):114–128, 1994.

[30] T. Kailath, S.-Y. Kung, and M. Morf. Displacement ranks of matrices and linear equations. *J. Math. Anal. and Appl.*, 68:395–407, 1979.

[31] T. Kailath and A. H. Sayed. Displacement structure: Theory and applications. *SIAM Review*, 37:297–386, 1995.

[32] T. Kailath, A. Vieira, and M. Morf. Inverses of Toeplitz operators, innovations, and orthogonal polynomials. *SIAM Rev.*, 20:106–119, 1978.

[33] N. Levinson. The Wiener RMS (root-mean-square) error criterion in filter design and prediction. *J. Math. Phys.*, 25:261–278, 1947.

[34] B. R. Musicus. Levinson and fast Cholesky algorithms for Toeplitz and almost Toeplitz matrices. Technical report, Research Lab. of Electronics, M. I. T., 1984.

[35] I. Schur. Uber potenzreihen, die in Innern des Einheitskrises Beschränkt Sind. *J. Reine Angew. Math.*, 147:205–232, 1917. English translation in: *I. Schur Methods in Operator Theory and Signal Processing*, I. Gohberg, ed., Birkhäuser, 1986, 31–89.

[36] G. Szegő. *Orthogonal Polynomials.* American Math. Soc., Providence, 1939.

[37] W. F. Trench. An algorithm for the inversion of finite Toeplitz matrices. *J. Soc. Indust. Appl. Math.*, 12:515–522, 1964.